14

Deep learning in steganography and steganalysis

Marc Chaumont[a,b]
[a] Montpellier University, LIRMM (UMR5506)/CNRS, Nîmes University, Montpellier, France
[b] LIRMM/ICAR, Montpellier, France

14.1 Introduction

Neural networks have been studied since the 1950s. Initially, they were proposed to model the behavior of the brain. In computer science, especially in artificial intelligence, they have been used for around 30 years for learning purposes. Ten or so years ago [1], neural networks were considered to have a lengthy learning time and to be less effective than classifiers such as SVMs or random forests.

With recent advances in the field of neuron networks [2], thanks to the computing power provided by graphics cards (GPUs), and because of the profusion of available data, deep learning approaches have been proposed as a natural extension of neural networks. Since 2012, these deep networks have profoundly marked the fields of signal processing and artificial intelligence, because their performances make it possible to surpass current methods and also to solve problems that scientists had not managed to solve until now [3].

In steganalysis, for the last 10 years, the detection of a hidden message in an image was mainly carried out by calculating rich models (RMs) [4] followed by a classification using the classifier EC [5]. In 2015 the first study using a convolutional neural network (CNN) obtained the first results of deep-learning steganalysis approaching the performances of the two-step approach (EC + RM[1]) [6]. During the period 2015–2018, many publications have shown that it is possible to an obtain improved performance in spatial steganalysis, JPEG steganalysis, side-informed steganalysis, quantitative steganalysis, and so on.

In Section 14.2, we present the structure of deep neural network generically. This section is focused on the existing publications in steganalysis and should be supplemented by reading about artificial learning and, in particular, gradient descent and stochastic gradient descent.

In three additional sections, not present in this chapter but available on ArXiv (https://arxiv.org/abs/1904.01444), we explain the different steps of the convolution module, tackle the complexity and learning times, and present the links between deep learning and previous approaches.

[1] We will note EC + RM to indicate the two-step approach based on the calculation of RMs and the use of EC.

Digital Media Steganography. https://doi.org/10.1016/B978-0-12-819438-6.00022-0

In Section 14.3, we revisit the different networks proposed during the period 2015–2018 for different scenarios of steganalysis. Finally, in Section 14.4, we discuss steganography by deep learning, which sets up a game between two networks in the manner of the precursor algorithm ASO [7].

14.2 The building blocks of a deep neuronal network

In the following subsections, we look back at the major concepts of a convolutional neural network (CNN). Specifically, we will recall the basic building blocks of a network based on the Yedroudj-Net[2] network, which was published in 2018 [8] (see Fig. 14.1), and which takes up the ideas present in Alex-Net [9], as well as the concepts present in networks developed for steganalysis including the very first network of Qian et al. [6], and the networks of Xu-Net [10], and Ye-Net [11].

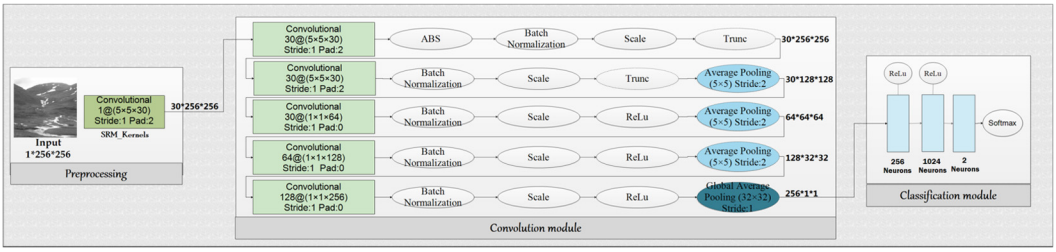

FIGURE 14.1 Yedroudj-Net network [8].

14.2.1 Global view of a Convolutional Neural Network

Before describing the structure of a neural network and its elementary components, it is useful to remember that a neural network belongs to the machine-learning family. In the case of supervised learning, which is the case that most concerns us, it is necessary to have a database of images with, for each image, its label, that is, its class.

Deep learning networks are large neural networks that can directly take raw input data. In image processing the network is directly powered by the pixels forming the image. Therefore a deep learning network learns in a joint way, both the compact intrinsic characteristics of an image (we speak of *feature map* or of *latent space*) and, at the same time, the separation boundary allowing the classification (we also talk of *separator plans*).

The learning protocol is similar to classical machine learning methods. Each image is given as an input to the network. Each pixel value is transmitted to one or more neurons. The network consists of a given number of *blocks*. A block consists of neurons that take real input values, perform calculations, and then transmit the actual calculated values to the next block. Therefore a neural network can be represented by an oriented graph

[2] GitHub link on Yedroudj-Net: https://github.com/yedmed/steganalysis_with_CNN_Yedroudj-Net.

where each node represents a computing unit. The learning is then completed by supplying the network with examples composed of an image and its label, and the network modifies the parameters of these calculation units (it learns) thanks to the mechanism of back-propagation.

The CNNs used for steganalysis are mainly built in three parts, which we will call *modules*: the preprocessing module, the convolution module, and the classification module. As an illustration, Fig. 14.1 schematizes the network proposed by Yedroudj et al. [8] in 2018. The network processes grayscale images of 256×256 pixels.

14.2.2 The preprocessing module

We can observe in Fig. 14.1 that in the *preprocessing module* the image is filtered by 30 high-pass filters. The use of one or more high-pass filters as preprocessing is present in the majority of networks used for steganalysis during the period 2015–2018.

An example of a kernel of a high-pass filter, the square S5a filter [4], is given in the equation

$$F^{(0)} = \frac{1}{12}\begin{pmatrix} -1 & 2 & -2 & 2 & -1 \\ 2 & -6 & 8 & -6 & 2 \\ -2 & 8 & -12 & 8 & -2 \\ 2 & -6 & 8 & -6 & 2 \\ -1 & 2 & -2 & 2 & -1 \end{pmatrix}. \tag{14.1}$$

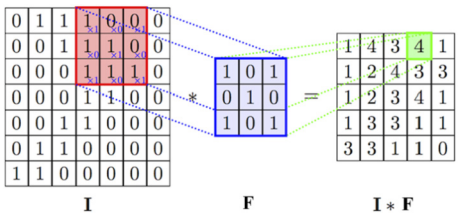

FIGURE 14.2 Principle of a convolution.

An illustration of the filtering (convolution) principle is given in Fig. 14.2. This preliminary filtering step allows the network to converge faster and is probably needed to obtain good performance when the learning database is too small [12] (only 4,000 pairs of cover/stego images of size 256×256 pixels). The filtered images are then transmitted to the first convolution block of the network. Note that the recent SRNet [13] network does not use any fixed prefilters, but learns the filters. It therefore requires a much larger database (more than 15,000 pairs of cover/stego images of size 256×256 pixels) and strong know-how for its initialization. Note that there is a debate in the community if one should use fixed filters or initialize the filters with prechosen values and then continue the learning, or learn filters with random initialization. At the beginning of 2019, in practice (real-world situation [14]) the best choice is probably in relation to the size of the learning database

(which is not necessary BOSS [15] or BOWS2 [16]) and the possibility to use transfer learning.

14.2.3 The convolution module

Within the *convolution module*, we find several macroscopic computation units, which we will call *blocks*. A *block* is composed of calculation units that take real input values, perform calculations, and return real values, which are supplied to the next block. Specifically, a *block* takes a set of *feature maps* (= a set of images) as input and returns a set of *feature maps* as output (= a set of images). Inside a block, there are a number of operations including the following four: the *convolution*, the *activation*, the *pooling*, and the *normalization* (details are given at https://arxiv.org/abs/1904.01444).

Note that the concept of neuron, as defined in the existing literature, before the emergence of convolutional networks, is still present, but it no longer exists as a data structure in neural network libraries. In convolution modules, we must imagine a neuron as a computing unit, which, for a position in the *feature map* taken by the convolution kernel during the convolution operation, performs the weighted sum between the kernel and the group of considered pixels. The concept of neuron corresponds to the scalar product between the input data (the pixels) and data specific to the neuron (the weight of the convolution kernel), followed by the application of a function from $\mathbb{R}$ in $\mathbb{R}$, called the activation function. Then by extension we can consider that pooling and normalization are operations specific to neurons.

Thus the notion of *block* corresponds conceptually to a "layer" of neurons. Note that in deep learning libraries, we call a *layer* any elementary operation such as convolution, activation, pooling, normalization, and so on. To remove any ambiguity, for the convolution module, we will talk about *block* and *operations*, and we will avoid using the term *layer*.

Without counting the preprocessing block, the *Yedroudj-Net* network [8] has a convolution module made of five convolution blocks, like the networks of Qian et al. [6] and Xu et al. [10]. The *Ye-Net* network [11] has a convolution module composed of 8 convolution blocks, and SRNet network [13] has a convolution module built with 11 convolution blocks.

14.2.4 The classification module

The last block of the convolution module (see the previous section) is connected to the *classification module*, which is usually a *fully connected* neural network composed of one to three blocks. This *classification module* is often a traditional neural network, where each neuron is fully connected to the previous *block* of neurons and to the next *block* of neurons.

The fully connected blocks often end with a softmax function, which normalizes the outputs delivered by the network between $[0, 1]$ so that the sum of the outputs equals one. The outputs are named imprecisely "probabilities". We will keep this denomination. So in the usual binary steganalysis scenario the network delivers two values as output: one giving the probability of classifying into the first class (e.g., the cover class) and the other

giving the probability of classifying into the second class (e.g., the stego class). The classification decision is then obtained by returning the class with the highest probability.

Note that in front of this *classification module*, we can find a *particular pooling* operation such as a *global average pooling*, a *spatial pyramid pooling (SPP)* [17], a *statistical moments extractor* [18], and so on. Such pooling operations return a fixed-size vector of values, that is, a feature map of fixed dimensions. The next block to this *pooling* operation is thus always connected to a vector of fixed size. So this block has a fixed input number of parameters. It is thus possible to present to the network images of any size without having to modify the topology of the network. For example, this property is available in the Yedroudj-Net [8] network, the Zhu-Net [19] network, or the Tsang et al. network [18].

Also note that [18] is the only paper, at the time of writing this chapter, that has seriously considered the viability of an invariant network to the dimension of the input images. The problem remains open. The solution proposed in [18] is a variant of the concept of average pooling. For the moment, there has not been enough studies on the subject to determine what is the correct topology of the network, how to build the learning data-base, how much the number of embedded bits influences the learning, or if we should take into account the *square root law* for learning at a fixed security level or any payload size, and so on.

14.3 The different networks used over the period 2015–2018

A chronology of the main CNNs proposed for steganography and steganalysis from 2015 to 2018 is given in Fig. 14.3. The first attempt to use deep learning methods for steganalysis

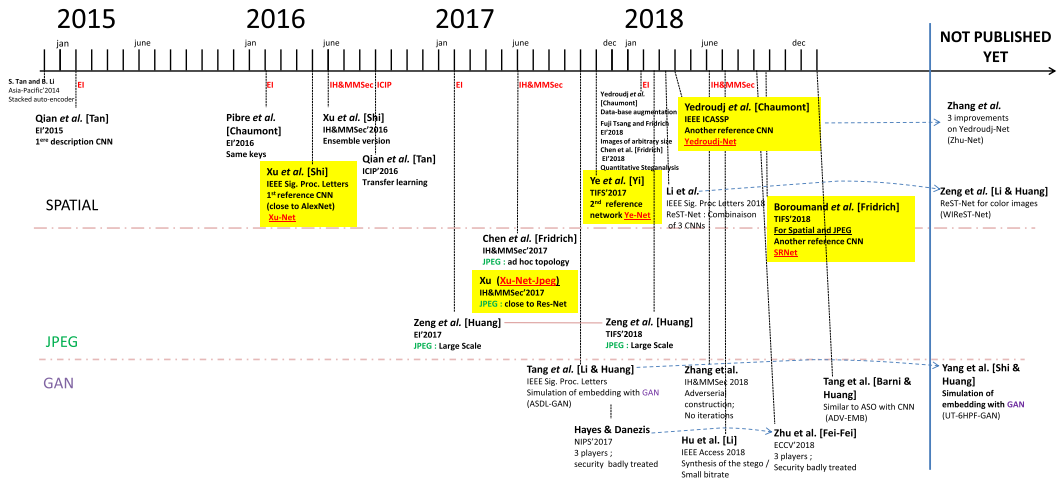

FIGURE 14.3 Chronology of the main CNNs for steganography and steganalysis from 2015 to 2018.

date back to the end of 2014 [20] with autoencoders. At the beginning of 2015, Qian et al. [6] proposed to use CNNs. One year later, Pibre et al. [21] proposed to pursue the study.

In 2016 the first results, close to those of current state-of-the-art methods (EC + RMs), were obtained with an ensemble of CNNs [22], as shown in Fig. 14.4. The Xu-Net[3] [10] CNN is used as a *base learner* of an ensemble of CNNs.

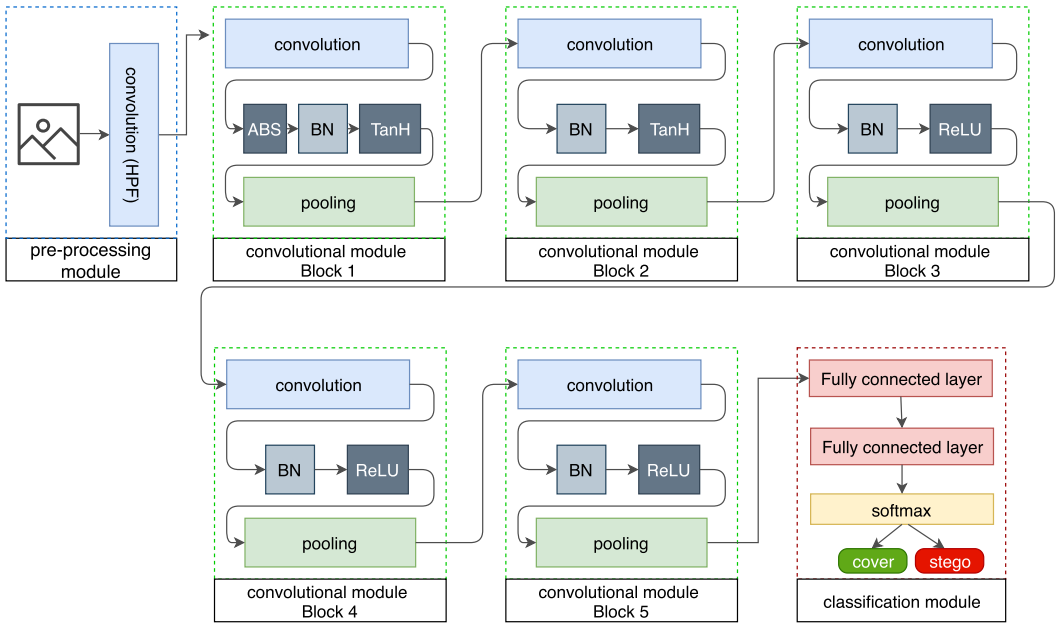

FIGURE 14.4 Xu-Net overall architecture.

Other networks were proposed in 2017, this time for JPEG steganalysis. In [23,24] (Figs. 14.5 and 14.6) the authors proposed a preprocessing inspired by RMs, and the use of a large learning database. The results were close to those of the existing state-of-the-art methods (EC + RMs). In [25] the network is built with a *phase-split* inspired by the JPEG compression process. An ensemble of CNNs was required to obtain results that were slightly better than those obtained by the current best approach. In Xu-Net-Jpeg [26] a CNN inspired by ResNet [27] with the *shortcut connection* trick and 20 blocks also improved the results in terms of accuracy. Note that in 2018 the ResDet [28] proposed a variant of Xu-Net-Jpeg [26] with similar results.

These results were highly encouraging, but regarding the gain obtained in other image processing tasks using deep learning methods [3], the steganalysis results represented less than a 10% improvement compared to the classical approaches that use an EC [5] with RMs

[3] In this chapter, we reference *Xu-Net* a CNN similar to the one given in [10], and not to the ensemble version [22].

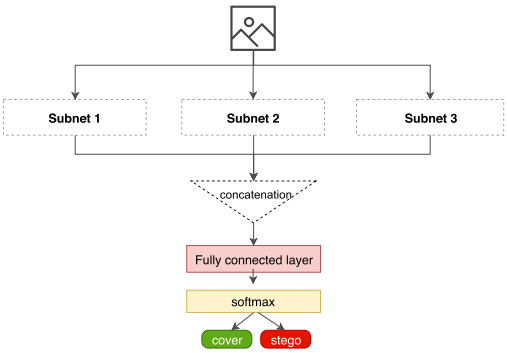

FIGURE 14.5 ReST-Net overall architecture.

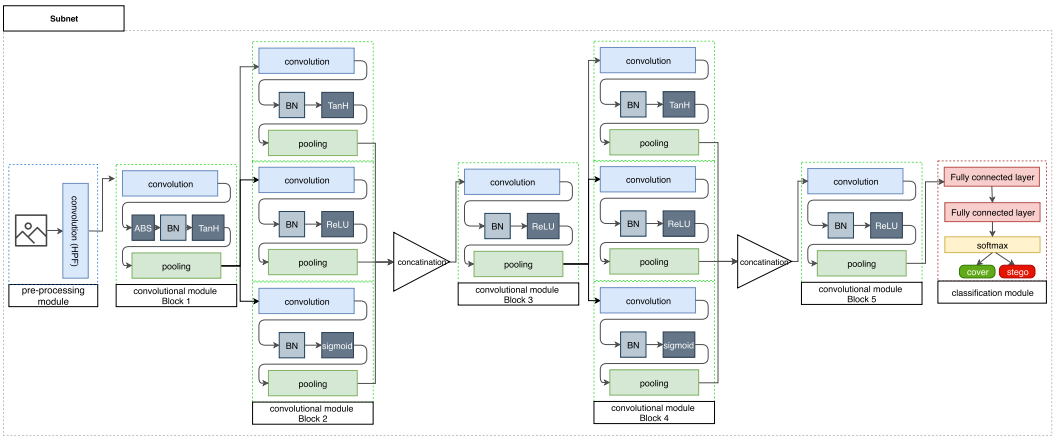

FIGURE 14.6 ReST-Net subnetwork.

[4,49] or RMs with a selection-channel awareness [29], [30], [31]. The revolutionary significant gain in the use of deep learning, observed in other areas of signal processing, was not yet present for steganalysis. In 2017 the main trends to improve CNN results used an ensemble of CNNs, modifying the topology by mimicking RMs extraction process or using ResNet. In most cases the design or the experimental effort was very high for a very limited improvement of performance in comparison to networks such as AlexNet [9], VGG16 [32], GoogleNet [33], ResNet [27], and so on, which inspired this research.

By the end of 2017 and early 2018 the studies had strongly concentrated on spatial steganalysis. Ye-Net [11] (Fig. 14.7), Yedroudj-Net[4] [12,8] (Fig. 14.8), ReST-Net [34] (Figs. 14.5 and 14.6), SRNet[5] [13] (Fig. 14.9) have been published respectively in November 2017,

[4] Yedroudj-Net source code: https://github.com/yedmed/steganalysis_with_CNN_Yedroudj-Net.
[5] SRNet source code: https://github.com/Steganalysis-CNN/residual-steganalysis.

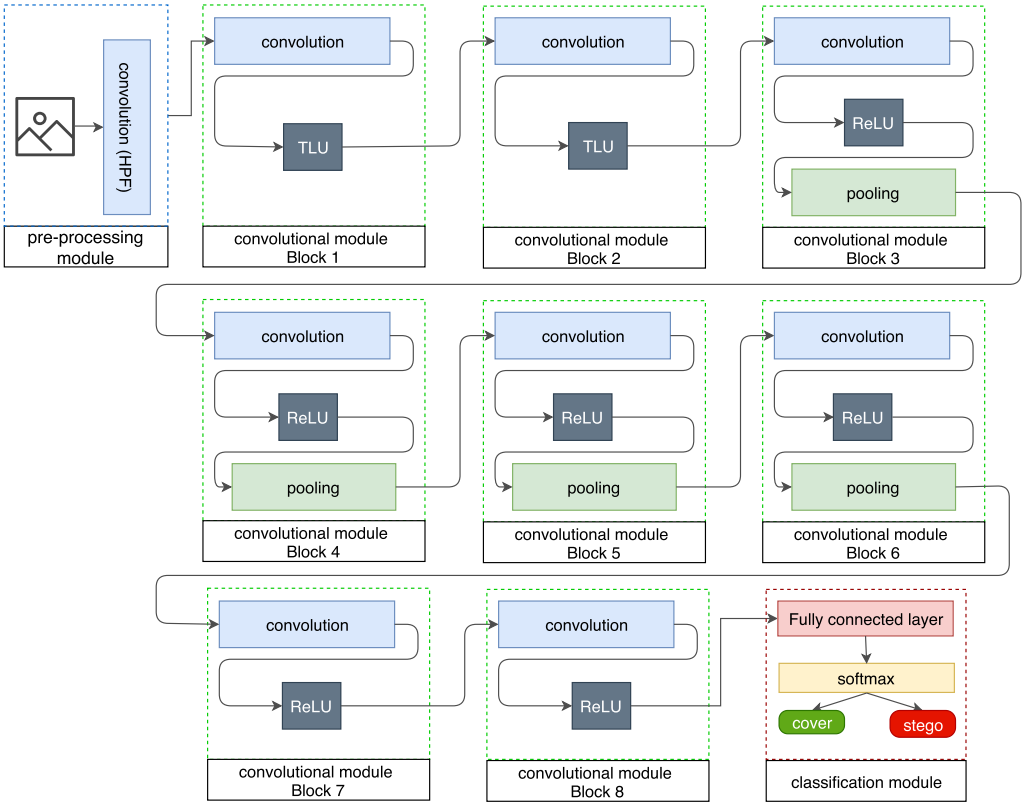

FIGURE 14.7 Ye-Net overall architecture.

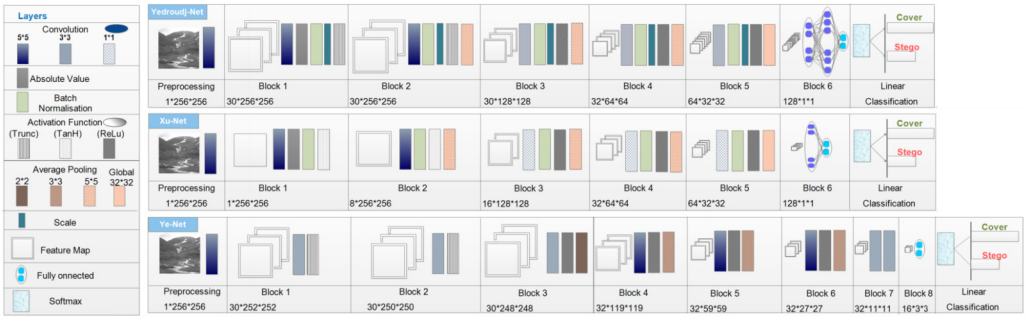

FIGURE 14.8 Comparison of Yedroudj-Net, Xu-Net, and Ye-Net architectures.

January 2018, May 2018, and May 2019 (with an online version in September 2018). All these networks clearly surpass the "old" two-step machine learning paradigm that used EC [5] and RMs [4]. Most of these networks can learn with a modest database size (i.e.,

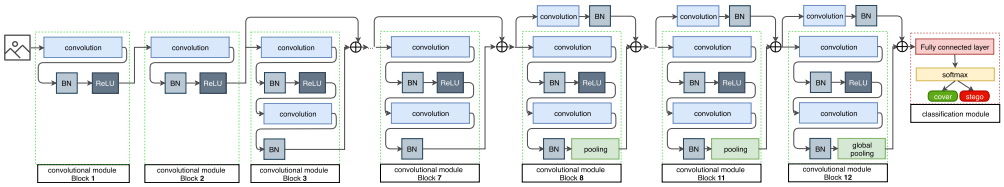

FIGURE 14.9 SRNet network.

around 15,000 pairs cover/stego of 8-bits-coded images of 256 × 256 pixels size from BOSS+BOWS2).

In 2018 the best networks were Yedroudj-Net [8], ReST-Net [34], and SRNet [13]. Yedroudj-Net is a small network that can learn on a very small database and can be effective even without using the tricks known to improve performance such as transfer learning [35], virtual augmentation of the database [11], and so on. This network is a good candidate when working on GANs. It is better than Ye-Net [11] and can be improved to face other more recent networks [19]. ReST-Net [34] is a huge network made of three subnetworks, which uses various preprocessing filter banks. SRNet [13] is a network that can be adapted to spatial or Jpeg steganalysis. It requires various tricks such as virtual augmentation and transfer learning and therefore requires a bigger database compared to Yedroudj-Net. These three networks are described in Section 14.3.1.

To resume, from 2015 to 2016, publications were in spatial steganalysis, and in 2017 the publications were mainly on JPEG steganalysis. In 2018, publications were again mainly concentrated on spatial steganalysis. Finally, at the end of 2017 the first publications using GANs appeared. In Section 14.4, we present new propositions using steganography by deep-learning and give classification per family.

In the next subsection, we report on the most successful networks until the end of 2018 for various scenarios. In Section 14.3.1, we describe the *not-side-channel-aware* (Not-SCA) scenario, in Section 14.3.2, we discuss the scenario known as *side-channel-aware* (SCA), and in Sections 14.3.3, we deal with JPEG steganalysis *Not-SCA* and *SCA* scenarios. In Section 14.3.4, we very briefly discuss cover-source mismatch, although for the moment, the proposals using a CNN do not exist.

We will not tackle the scenario of CNN invariant to the size of the images because it is not yet mature enough. This scenario is briefly discussed in Section 14.2.4, and the papers of Yedroudj-Net [8], Zhu-Net [19], or Tsang et al. [18] give the first solutions.

We will not approach the scenario of quantitative steganalysis per CNN, which consists in estimating the embedded payload size. This scenario is very well examined in the paper [36] and serves as a new state-of-the-art method. The approach surpasses the previous state-of-the-art approaches [37,38] that rely on RMs, an ensemble of trees, and an efficient normalization of features.

Nor will we discuss batch steganography and pooled steganalysis with CNNs, which has not yet been addressed, although the work presented in [39] using two-stage machine learning can be extended to deep learning.

14.3.1 The spatial steganalysis Not-Side-Channel-Aware (Not-SCA)

In early 2018 the most successful spatial steganalysis approach is the Yedroudj-Net [8] method (Fig. 14.7). The experimental comparisons were carried out on the BOSS database, which contains 10,000 images subsampled to 256×256 pixels. For a fair comparison, the experiments were performed by comparing the approach to Xu-Net without EC [10] to the Ye-Net network in its Not-SCA version [11], and also to EC [5] fed by spatial RMs [4]. Note that Zhu-Net [19] (not yet published when writing this chapter) offers three improvements to Yedroudj-Net, which allows it to be even more efficient. The improvements reported by Zhu-Net [19] are the update to the kernel filters of the preprocessing module (in the same vein as what has been proposed by Matthew Stamm's team in Forensics [40]), replacing the first two convolution blocks with two modules of *depthwise separable convolutions* as proposed in [41], and finally replacing global average pooling with a *spatial pyramid pooling (SPP)* module as in [17].

In May 2018 the ReST-Net [34] approach was proposed (see Figs. 14.5 and 14.6). It consists of agglomerating three networks to form a *supernetwork*. Each subnet is a modified Xu-Net like network [10] resembling the Yedroudj-Net [8] network, with an Inception module on blocks 2 and 4. This Inception module contains filters of the same size, with a different activation function for each "path" (TanH, ReLU, Sigmoid). The first subnet performs preprocessing with 16 Gabor filters, the second subnetwork performs preprocessing with 16 SRM linear filters, and the third network performs preprocessing with 14 nonlinear residuals (min and max calculated on SRM). The learning process requires four steps (one step per subnet and then one step for the *supernetwork*). The results are 2–5% better than Xu-Net for S-UNIWARD [42], HILL [43], CMD-HILL [44] on BOSSBase v1.01 [15] 512×512. Looking at the results, it is the concept of ensemble that improves the performances. Taken separately, each subnet has a lower performance. At the moment, no comparison in a fair framework was made between an ensemble of Yedroudj-Net and ReST-Net.

In September 2018 the SRNet [13] approach became available online (see Fig. 14.9). It proposes a deeper network than previous networks, which is composed of 12 convolution blocks. The network does not perform preprocessing (the filters are learned) and subsamples the signal only from the 8th convolution block. To avoid the problem of vanishing gradient, blocks 2–11 use the shortcut mechanism. The Inception mechanism is also implemented from block 8 during the pooling (subsampling) phase. The learning database is augmented with the BOWS2 database as in [11] or [12], and a curriculum training mechanism [11] is used to change from a standard payload size of 0.4 bpp to other payload sizes. Finally, gradient descent is performed by Adamax [45]. The network can be used for spatial steganalysis (Not-SCA), informed (SCA) spatial steganalysis (As discussed in Section 14.3.2), and JPEG steganalysis (see Section 14.3.3, Not-SCA or SCA). Overall the philosophy remains similar to the previous networks, with three parts: preprocessing (with learned filters), convolution blocks, and classification blocks. With a simplified vision, the network corresponds to the addition of 5 blocks of convolution without pooling, just after the first convolution block of Yedroudj-Net network. To be able to use this large number of blocks on a modern GPU, the authors must reduce the number of feature maps to 16,

and to avoid the problem of vanishing gradients, they must use the trick of residual shortcut within the blocks as proposed in [27]. Note that preserving the size of the signal in the first seven blocks is a radical approach. This idea has been put forward in [21], where the suppression of pooling had clearly improved the results. The use of modern brick like shortcuts or Inception modules also enhances performance.

It should also be noted that the training is completed end-to-end without particular initialization (except when there is a curriculum training mechanism). In the initial publication [13], SRNet network was not compared to Yedroudj-Net [8] or to Zhu-Net [19], but later, in 2019, in [19] all these networks have been compared, and the update of Yedroudj-Net, that is, Zhu-Net gives performances of 1% to 4% improvement over SRNet, and 4% to 9% improvement over Yedroudj-Net, when using the usual comparison protocol. Note that Zhu-Net is also better than the network *Cov-Pool* published at IH&MMSec'2019 [46], and its performances are similar to SRNet.

14.3.2 The spatial steganalysis Side-Channel-Informed (SCA)

At the end of 2018, two approaches combined the knowledge of the selection channel, the SCA-Ye-Net (which is the SCA version of Ye-Net) [11] and the SCA-SRNet (which is the SCA version of SRNet) [13]. The idea is using a network for non-informed steganalysis and injecting not only the image to be steganalyzed, but also the modification probability map. It is thus assumed that Eve knows or can have a good estimation [47] of the modification probability map, that is, Eve has access to side-channel information.

The modification probability map is given to the preprocessing block SCA-Ye-Net [11] and equivalently to the first convolution block for SCA-SRNet [13], but the kernel values are replaced by their absolute values. After the convolution, each feature map is summed pointwise with the corresponding convolved "modification probability map" (Fig. 14.10). Note that the activation functions of the first convolutions in SCA-Ye-Net, that is, the trun-

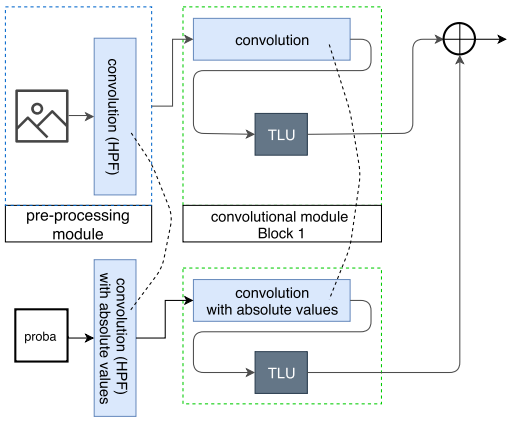

FIGURE 14.10 Integration of the modification probability map in a CNN.

cation activation function (*truncated linear unit (TLU)* in the paper), are replaced by a ReLU. This makes it possible to propagate (forward pass) "virtually" throughout the network, an information related to the image, and another related to the modification probability map.

Note that this procedure to transform a Not-SCA-CNN into an SCA-CNN is inspired by the propagation of the modification probability map proposed in [30] and [31]. These two papers come as an improvement on the previous maxSRM RMs [29]. In maxSRM, instead of accumulating the number of occurrences in the cooccurrence matrix, an accumulation of the maximum of a local probability was used. In [30] and [31] the idea was transforming the modification probability map in a similar way to the filtering of the image, and then to updating the cooccurrence matrix using the transformed version of the modification probability map instead of the original modification probability map. The imitation of this principle was initially integrated into Ye-Net for CNN steganalysis, and this concept is easily transposable to most of the modern CNNs.

14.3.3 The JPEG steganalysis

The best JPEG CNN at the end of 2018 was SRNet [13]. Note that this network, at this period, is the only one that has been proposed with a Side Channel Aware (SCA) version.

It is interesting to list and rapidly discuss the previous CNNs used for JPEG steganalysis. The first network, published in February 2017, was the Zeng et al. network and was evaluated with a million images and does a limited evaluation of stego-mismatch [23,24]. Then in June 2017 at IH&MMSec'2017, two networks have been proposed, PNet [25] and Xu-Net-Jpeg [26]. Finally, SRNet [13] was added online in September 2018.

In Zeng et al.s' network [23,24] the preprocessing block takes as input a de-quantized (real value) image, then convolved it with 25 DCT basis, and then quantized and truncated the 25 filtered images. This preprocessing block uses handcrafted filter kernels (DCT basis), the kernels' values are fixed, and these filters are inspired by DCTR RMs [48]. There are three different quantizations, so, the preprocessing block gives 3×25 residual images. The CNN is then made of three subnetworks, each producing a feature vector of dimension 512. The subnetworks are inspired by Xu-Net [10]. The three feature vectors are outputted by the three subnetworks and then given to a fully connected structure, and the final network ends with a softmax layer.

Similarly to what has been done for spatial steganalysis, this network is using a preprocessing block inspired by RMs [48]. Note that the most efficient RMs today is the Gabor filter RMs [49]. Also, note that this network takes advantage of the notion of an ensemble of features, which comes from the three different subnetworks. The network of Zeng et al. is less efficient than Xu-Net-Jpeg [26] but gives an interesting first approach guided by RMs.

The main PNet idea (and also VNet, which is less efficient but takes less memory) [25] is to imitate phase-aware RMs, such as DCTR [48], PHARM [50], or GFR [49], and therefore to have a decomposition of an input image into 64 features maps, which represents the 64 phases of the Jpeg images. The pre-processing block takes as input a dequantized (real-valued) image, convolves it with four filters, the "SQUARE5$\times$5" from the Spatial Rich

Models [4], a "point" high-pass filter (referenced as "catalyst kernel"), which complements the "SQUARE5×5", and two directional Gabor filters (angles 0 and $\pi/2$).

Just after the second block of convolution, a "PhaseSplit Module" splits the residual image into 64 feature maps (one map = one phase), similarly to what was done in RMs. Some interesting methods have been used such as (1) the succession of the fixed convolutions of the preprocessing block, and a second convolution with learnable values, (2) a clever update of BN parameters, (3) the use of the "Filter Group Option", which virtually builds subnetworks, (4) bagging on 5-cross-validation, (5) taking the 5 last evaluations to give the mean error for a network, (6) shuffling the database at the beginning of each epoch to have better BN behavior and to help generalization, and (7) eventually using an Ensemble. With such know-how, PNet beat the classical two-step machine learning approaches in a Not-SCA and also in an SCA version (EC + GFR).

The Xu-Net-Jpeg [26] is even more attractive since the approach was slightly better than PNet and does not require a strong domain inspiration like in PNet. The Xu-Net-Jpeg is strongly inspired by ResNet [27], a well-established network from the machine learning community. ResNet allows the use of deeper networks thanks to the use of shortcuts. In Xu-Net-Jpeg the preprocessing block takes as an input a dequantized (real-valued) image, then convolves the image with 16 DCT bases (in the same spirit as Zeng et al. network [23,24]) and then applies an absolute value, a truncation, and a set of convolutions, BN, ReLU until it obtains a feature vector of dimension 384, which is given to a fully connected block. Note that the max pooling or average pooling are replaced by convolutions. This network is really simple and in 2017 was the state-of-the-art method. In a way, this kind of results shows us that the networks proposed by machine learning community are very competitive, and there is no so much domain-knowledge to integrate to the topology of a network to obtain a very efficient network.

In 2018 the state-of-the-art CNN for JPEG steganalysis (which can also be used for spatial steganalysis) was SRNet [13]. This network was previously presented in Section 14.3.1. Note that for the side channel aware version of SRNet, the *embedding change probability* per DCTs coefficient is first mapped back in the spatial domain using absolute values for the DCT basis. This *side-channel map* then enters the network and is convolved with each kernel (this first convolution acts as a preprocessing block). Note that the convolutions in this first block for this *side-channel* map are such that the filter kernels are modified to their absolute values. After passing the convolution, the feature maps are summed with the square root of the values from the convolved *side-channel* map. Note that this idea is similar to that exposed in SCA Ye-Net version (SCA-TLU-CNN) [11] about the integration of a side-channel map, and to the recent proposition for side-channel-aware steganalysis in JPEG with RMs [31], where the construction of the *side-channel* map and especially the quantity $\delta_{\text{uSA}}^{1/2}$ [6] was defined.

Note that a similar solution with more convolutions, applied to the *side-channel map*, have been proposed in IH&MMSec'2019 [51].

[6] uSA stands for Upper bounded Sum of Absolute values.

14.3.4 Discussion about the Mismatch phenomenon scenario

Mismatch (cover-source mismatch or stego-mismatch) is a phenomenon present in machine learning, and this issue sees decrease of classification performances because of the inconsistency between the distribution of the learning database and the distribution of the test database. The problem is not due to an inability to generalize machine learning algorithms, but due to the lack of similar examples occurring in the training and test database. The problem of mismatch is an issue that goes well beyond the scope of steganalysis.

In steganalysis the phenomenon can be caused by many factors. The cover-source mismatch can be caused by the use of different photosensors, by different digital processing, by different camera settings (focal length, ISO, lens, etc.), by different image sizes, by different image resolutions, and so on [52,53]. The stego-mismatch can be caused by different amounts of embedded bits or by different embedding algorithms.

Even if not yet fully explored and understood, the mismatch (cover-source mismatch (CSM) or stego mismatch) is a major area for examination in the coming years for the discipline. The results of the Alaska challenge [54][7] published at the ACM conference IH&MMSec'2019 will continue these considerations.

In 2018, CSM had been established for 10 years [55]. There are two major current schools of thought, as well as a third more exotic one:

- The first school of thought is the so-called *holistic* approach (that is, global, macroscopic, or systemic) and consists of learning all distributions [56,57]. The use of a single CNN with millions of images [24] is in the logical continuation of this current school of thought. Note that this scenario does not consider that the test set can be used during learning. This scenario can be assimilated to an *online scenario*, where the last player (from a game theory point of view) is the steganographer because in an online scenario the steganographer can change her strategy, whereas the steganalyzer cannot.
- The second school of thought is *atomistic* (= partitioned, microscopic, analytical, of divide-and-conquer type, or individualized) and consists of partitioning the distribution [58], that is, creating a partition and associating a classifier for each cell of the partition. Note that an example of an atomistic approach for stego-mismatch management, using a CNN multiclassifier, is presented in [59] (a class is associated with each embedding algorithm, and thus there is a latent partition). Note that this idea [59], among others, has been used by the winners of the Alaska challenge [60]. Note that again, this scenario does not consider that the test set can be used during learning. This scenario can also be assimilated to an *online scenario* where the last player (from a game theory point of view) is the steganographer, because in an online scenario the steganographer can change her strategy, whereas the steganalyst cannot.
- Finally, the third exotic school of thought considers that there is a test database (with much more than one image) and that the database is available and usable (without labels) during learning. This scenario can be assimilated to an *offline scenario* where

[7] Alaska: A challenge of steganalysis into the wilderness of the real world. https://alaska.utt.fr/.

the last player (from a game theory point of view) is the steganalyzer, because in this offline scenario the steganalyzer is playing a more forensic role. In this situation, there are approaches of type domain adaptation, or a transfer of features GTCA [61], IMFA [62], CFT [63], where the idea is defining an invariant latent space. Another approach is ATS [64], which performs an unsupervised classification using only the test database and requires the embedding algorithm to reembed a payload in the images from the test database.

These three schools of thought can help derive approaches by CNN that integrate the ideas presented here. That said, the ultimate solution may be detecting the phenomenon of mismatch and raising the alarm or prohibiting the decision [65]. In short, integrating a more intelligent mechanism than just holistic or atomistic.

14.4 Steganography by deep learning

In Simmons' founding paper [66], steganography and steganalysis are defined as a *3-player game*. The steganographers, usually named Alice and Bob, want to exchange a message without being suspected by a third party. They must use a harmless medium, such as an image, and hide the message in this medium. The steganalyst, usually called Eve, observes the exchanges between Alice and Bob. Eve must check whether these images are natural, that is, cover images, or whether they hide a message, that is, stego images.

This notion of *game* between Alice, Bob, and Eve corresponds to that found in game theory. Each player tries to find a strategy that maximizes his chance of winning. For this, we express the problem as a min–max problem that we seek to optimize. The solution to the optimum, if it exists, is called the solution at the Nash equilibrium. When all the players are using a strategy at the Nash equilibrium, any change of strategy from a player leads to a counterattack from the other players allowing them to increase their gains.

In 2012, Schöttle and Böhme [67,68] have modeled with a simplifying hypotheses a problem of steganography and steganalysis and proposed a formal solution. Schöttle and Böhme called this approach the *optimum adaptive steganography* or *strategic adaptive steganography* in opposition to the so-called *naive adaptive steganography*, which corresponds to what is currently used in algorithms like HUGO (2010) [69], WOW (2012) [70], S-UNIWARD / J-UNIWARD / SI-UNIWARD (2013) [42], HILL (2014) [43], MiPOD (2016) [71], Synch-Hill (2015) [72], UED (2012) [73], IUERD (2016) [74], IUERD-*UpDist-Dejoin2* (2018) [75], and so on.

A mathematical formalization of the steganography/steganalysis problem by game theory is difficult and often far from practical in reality. Another way to determine a Nash equilibrium is "simulating" the game. From a practical point of view, Alice plays the entire game alone, meaning that she does not interact with Bob or Eve to build her embedding algorithm. The idea is that she uses three algorithms (two algorithms in the simplified version) that we name *agents*. Each agent plays the role of Alice, Bob,[8] and Eve, and each

[8] Bob is deleted in the simplified version.

agent runs at Alice's home. Note these three algorithms running at Alice's home: *Agent-Alice*, *Agent-Bob*, and *Agent-Eve*. With these notations, we thus make a distinction with the human users: Alice (sender), Bob (receiver), and Eve (warden), and it allows us to highlight the fact that the three agents are executed from Alice's side. So, Agent-Alice's role is to embed a message into an image so that the resulting stego image is undetectable by Agent-Eve and such that Agent-Bob can extract the message.

Alice can launch the game, that is, the simulation, and the agents are "fighting".[9] Once the agents have reached a Nash's equilibrium, Alice stops the simulation and can now keep Agent-Alice, which is her *strategic adaptive embedding* algorithm, and can send Agent-Bob, that is, the extraction algorithm (or any equivalent information) to Bob.[10] The secret communication between Alice and Bob is now possible through the use of the Agent-Alice algorithm for embedding and Agent-Bob algorithm for extraction.

The first precursor approaches aimed at simulating a *strategic adaptive equilibrium* and therefore proposing *strategic embedding* algorithms from 2011 and 2012. The two approaches are MOD [76] and ASO [7,77], as shown in Fig. 14.11. Whether for MOD or ASO, the game is made by pitting Agent-Alice and Agent-Eve against each other. In this game, Agent-Bob is not used since Agent-Alice is simply generating a cost map, which is then used for coding and embedding the message thanks to an STC [78]. Alice can generate a cost map for a source image with the Agent-Alice, and then she can easily use the STC [78] algorithm to embed her message and obtain the stego image. From his side, Bob only has to use the STC [78] algorithm to retrieve the message from the stego image.

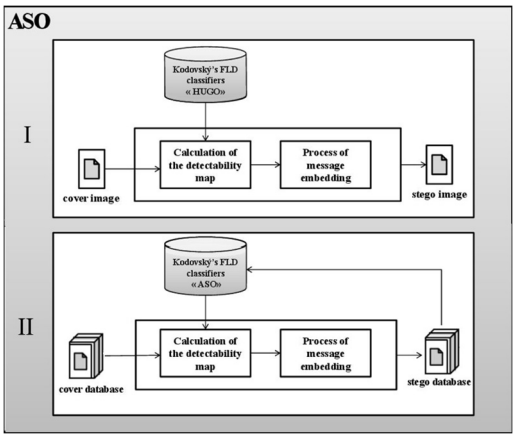

FIGURE 14.11 General scheme of ASO [7,77].

[9] The reader should be aware that from a game theory point of view, there are only two teams that are competing (Agent-Alice plus Agent-Bob from one side and Agent-Eve from the other) in a zero-sum game.

[10] Note that the exchange of any secret information between Alice and Bob, prior to the use of Agent-Alice and Agent-Bob, requires the use of another steganographic channel. Also note that this initial sending from Alice to Bob before been able to use Agent-Alice and Agent-Bob is equivalent to the classical stego-key exchange problem.

In both MOD or ASO, the "simulation" is such that the following two actions are iterated until a stop criterion is reached:

i) Agent-Alice updates its embedding cost map by asking an Oracle (the Agent-Eve) how best to update each embedding cost to be even less detectable.

In MOD (2011) [76], Agent-Eve is an SVM. Agent-Alice updates their embedding costs by reducing the SVM margin separating the covers and the stegos.

In ASO (2012) [7], Agent-Eve is an EC [5] and is named an Oracle. Agent-Alice updates the embedding costs by transforming a stego into a cover.

In both cases the idea is finding a displacement in the latent space (feature space) colinear to the orthogonal axis to the hyperplane separating the cover and stego classes. Note that in the current terminology, introduced by Ian Goodfellow in 2014 [79], Agent-Alice runs an adversarial attack, and the Oracle (Agent-Eve), called a discriminator (or the classifier to be deceived), must learn to counter this attack.

ii) The Oracle (Agent-Eve) updates its classifier. Reformulated with the terminology from machine learning, this equates to the discriminant update by relearning it to steganalyze once more the stego images generated by Agent-Alice.

In 2014, Goodfellow et al. [79] used neural networks to "simulate" a game with an *image generator network* and a *discriminating network* whose role was to decide whether an image was real or synthesized. The authors have named this *generative adversarial networks* (GAN approach). The terminology used in this paper was subsequently widely adopted. Moreover, the use of neuron networks makes the expression of the min–max problem easy. The optimization is then carried out via the back-propagation optimization process. Moreover, thanks to deep-learning libraries, it is now easy to build a GAN-type system. As we have already mentioned before, the concept of game simulation existed in steganography/steganalysis with MOD [76] and ASO [7], but the implementation and optimization become easier with neural networks.

From 2017, after a period of 5 years of stagnation, the concept of the simulated game is once again studied in the field of steganography/steganalysis, thanks to the emergence of deep learning and GAN approaches. At the end of 2018, we can define four groups or four families[11] of approaches; some of them will probably merge:

- The family by synthesis;
- The family by generation of the modifications probability map;
- The family by adversarial-embedding *iterated* (approaches misleading a discriminant);
- The family by 3-player game.

[11] "Deep Learning in Steganography and Steganalysis since 2015", tutorial given at the "Image Signal & Security Mini-Workshop", the 30th of October 2018, IRISA/Inria Rennes, France, DOI: 10.13140/RG.2.2.25683.22567, *http://www.lirmm.fr/~chaumont/publications*. Look at the slides (http://www.lirmm.fr/~chaumont/publications/Deep_Learning_in_Steganography_and_Steganalysis_since_2015_Tutorial_Meeting-France-CHAUMONT_30_10_2018.pdf) and the video of the talk (https://videos-rennes.inria.fr/video/H1YrIaFTQ).

14.4.1 The family by synthesis

The first approaches based on *image synthesis* via a GAN [79] generator proposed the generation of cover images and then using them to make insertion by modification. These early propositions were approaches *by modification*. The argument put forward for such approaches is that the generated database would be safer. A reference often cited is that of SGAN [80] found on ArXiv, which was rejected at ICLR'2017 and was subsequently never published. This unpublished paper has a lot of errors and lack of proof. We should rather prefer the reference of SSGAN [81] published in September 2017, which proposes the same thing: generating images and then hiding messages in them. However, this protocol seems to complicate the matter. It is more logical that Alice herself chooses natural images that are safe for embedding, that is, images that are innocuous, never broadcasted before, adapted to the context, with lots of noise or textures [82], not well classified by a classifier [77], or with a small deflection coefficient [71], rather than generating images and then using them to hide a message.

A much more interesting approach using *synthesis* is to directly generate images that will be considered stego. To my knowledge, the first approach exploiting the GAN mechanism for image synthesis using the principle of steganography *without modifications* [83] is proposed in the paper of Hu et al. [84] and published in July 2018; see Fig. 14.12.

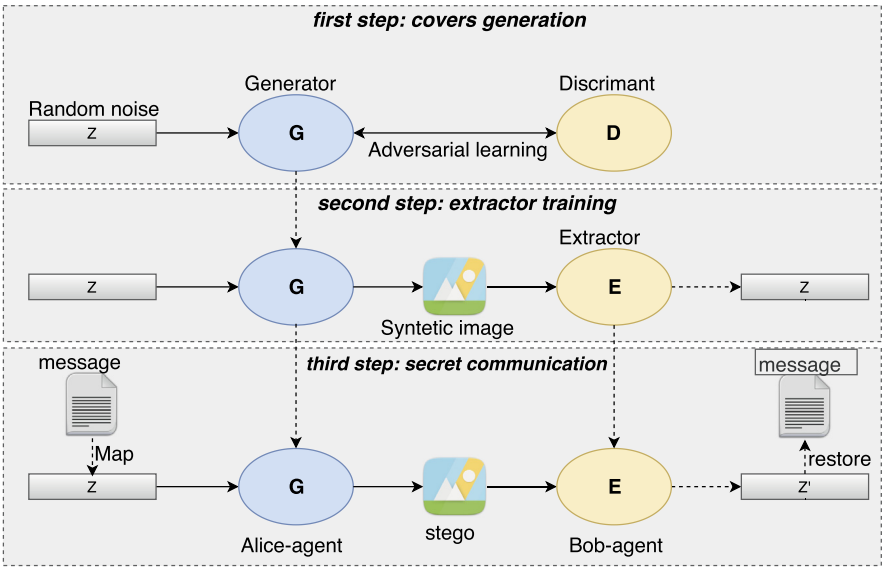

FIGURE 14.12 Hu et al. [84] approach by synthesis without modification.

The first step consists of deriving a network able to synthesize images. In this paper the DCGAN generator [85] is used to synthesize images with a preliminary learning thanks to GAN methodology. When fed with a vector of a fixed-size uniformly distributed in $[-1, 1]$, the generator synthesizes an image. The second step consists of learning another network

to extract a vector from a synthesized image; the extracted vector must correspond to the vector given at the input of the generator that synthesizes the image. Finally, the last step consists of sending to Bob the extraction network. Now Alice can map a message to a fixed-size uniformly distributed vector and then synthesize an image with the given vector and send it to Bob. Bob can extract the vector and retrieve the corresponding message.

The approaches with *no modifications* have been around for many years, and it is known that one of the problems is that the number of bits that can be communicated is lower compared to the approaches with modifications. That said, the gap between the approaches by *modifications* versus *no-modifications* is beginning to narrow.

Here is a rapid analysis of the efficiency of the method. In the paper of Hu et al. [84] the capacity is around 0.018 bits per pixel (bpp) with images of 64×64 pixels.[12] In the experiment carried out the synthesized images are either faces or photos of food. An algorithm like HILL [43] (one of the most powerful algorithms on the BOSS database [82]) is detected by SRNet [13] (one of the most successful steganalysis approaches toward the end of 2018) with error probability $Pe = 31.3\%$ (note that Pe of 50% is equivalent to a random detector) on a 256×256 pixel BOSS database for a payload size of 0.1 bpp. Due to the square root law, the Pe would be higher for the 64×64 pixel BOSS database.

Therefore there is around 0.02 bpp for the unmodified synthetic approach of Hu et al. [84], whose security has not yet been evaluated enough, against something around 0.1 bpp for HILL with less than one chance in three to be detected with a *clairvoyant* steganalysis, that is, a laboratory steganalysis (unrealistic and much more efficient than a "real-world"/"into the wild" steganalysis [14,54]). Therefore there is still a margin in terms of the number of bits transmitted between the *no-modification* synthesis-based approaches, such as that of Hu et al. approach [84], and *modification* approaches such as S-UNIWARD [42], HILL [43], MiPod [71], or even Synch-Hill [72], but this margin has been reduced.[13] Also, note that there are still some issues to be addressed to ensure that approaches such as that proposed by Hu et al. are entirely safe. In particular, it must be ensured that the detection of synthetic images [86] does not compromise the communication channel in the long term. It must also be ensured that the absence of a secret key does not jeopardize the approach. Indeed, if one considers that the generator is public, then is it possible to use this information to deduce that a synthesis approach without modification has been used?

14.4.2 The family by generation of the modifications probability map

The family by generation of the modification probability map is summarized in the late 2018s in two papers: ASDL-GAN [87] and UT-6HPF-GAN [88]; see Fig. 14.13. In this ap-

[12] The vector dimension is 100. This vector is used to synthesize images of a size $64 \times 64 \times 3$. There are 100×3 bits (see the mapping) per image, i.e. about 0.02 bits per pixel (bpp). The Bit Error Rate is $BER = 1 - 0.94 = 6\%$. It is, therefore, necessary to add an Error Correcting Code (ECC) so that the approach is without errors. With the use of a Hamming code [15, 11, 3] that corrects at best 6% of errors, the payload size is therefore around 0.018 bpp.

[13] The other families of steganography by deep learning, which are *modification* based, will probably help to maintain this performance gap for a few years more.

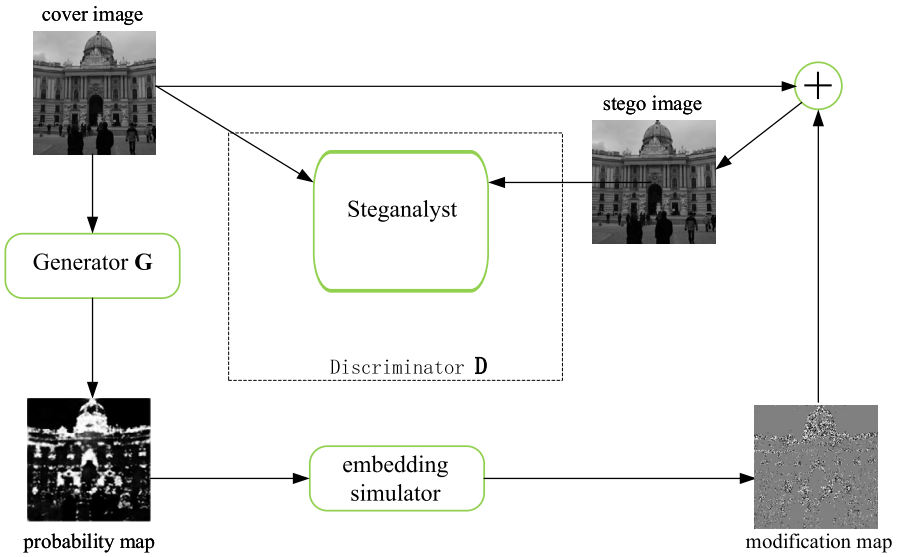

FIGURE 14.13 ASDL approach; generation of the modifications probability map.

proach, there is a generator network and a discriminant network. From a cover the generator network generates a map, which is called the modification probability map. This modification probability map is then passed to an equivalent of the random draw function used in the STC [78] simulator. We then obtain a map whose values belong to $\{-1, 0, +1\}$. This map is called the modification map and corresponds to the so-called stego-noise. The discriminant network takes as input a cover or an image resulting from the summation (point-to-point sum) of the cover and the stego-noise generated by the generator. The discriminant's objective is to distinguish between the cover and the "*cover* + *stego-noise*" image. The generator's objective is to generate a modification map that makes it possible to mislead the discriminant the most. Of course, the generator is forced to generate a non-zero probability map by adding in the loss term a term constraining the size of the payload in addition to the term misleading the discriminant.

In practice, taking the latest approach UT-6HPF-GAN [88] the generator is a U-Net type network, the draw function is obtained by a differentiable function *double Tanh*, and the discriminant is the Xu-Net [10] enriched with six high-pass filters for the preprocessing in the same spirit as Ye-Net [11] or Yedroudj-Net [8].

The system learns on a first database, and then security comparisons are made on the 256×256 pixel BOSS [15], LIRMMBase [21], and BOWS2 [16] databases. The steganalysis is done with the EC [5] fed by SRM [4], with EC plus the MaxSRM [29], and with Xu-Net [10]. Note that using Xu-Net is not a good choice since it is less efficient than EC+SRM or EC+MaxSRM and also because it is the discriminant in the UT-6HPF-GAN (there is a risk of falling into an "incompleteness" issue; see [89,90]). So, only looking at the results with EC+SRM on the BOSS database, with real embedding using STC [78], the performances are

equivalent to those of HILL [43], which is one of the most efficient embedding algorithms on BOSS [82]. It is therefore a very promising family.

Additionally, the generator does not seem to be impacted when used on a database that is different from the learning database. Nevertheless, curriculum learning has to be used when the target payload is changed, which seems to indicate a kind of sensitivity to the mismatch. Further reflexions have also to be achieved related to the generator's loss and to the mixing of both a security-related term and a payload-size term. Usually, one of the two criteria is fixed, so that we have to be in a payload-limited sender scenario or a security-limited sender scenario. Note that a version for JPEG has been proposed in IH&MMSec'2019, JS-GAN [91].

14.4.3 The family by adversarial-embedding *iterated* (approaches misleading a discriminant)

The family by adversarial-embedding *iterated* reuses the concept of *game simulation*, which was presented in the beginning of Section 14.4 with a simplification of the problem since there are only two players, Agent-Alice and Agent-Eve. Historically, MOD [76] and ASO [7] were the first algorithms of this type.

Recently, some papers have used the adversarial concept[14] by generating a deceiving example (see [92]), but these approaches are not adversarial-embedding *iterated*, nor they are dynamic: they contain no game simulation, they do not try to reach a Nash equilibrium, and there is no learning alternation between the embedder and the steganalysis.

A paper with spirit more in tune with a simulation of a game, which takes the principle of ASO [7] and whose objective is updating the cost map, is the algorithm ADV-EMB [93] (previously called AMA in *ArXiv* arXiv:1803.09043). In this paper the authors propose to make an adversarial-embedding *iterated*, by letting Agent-Alice access the gradient of the loss of Agent-Eve (similarly to ASO, where Agent-Alice has access to its Oracle (the Agent-Eve)). In ADV-EMB, Agent-Alice uses the gradient of the direction to the class frontier (between classes cover and stego) to modify the cost map, and in ASO, Agent-Alice directly uses the direction of the class frontier to modify the cost map.

In ADV-EMB [93] the cost map is initialized with the cost of S-UNIWARD (for ASO, it was the cost of HUGO [69]). During the iterations, the cost map is updated, but there is only a β percentage of values that are updated.[15] When the ADV-EMB iterations are stopped, the cost map is composed of a $\beta - 1$ percent of positions having a cost defined by S-UNIWARD and β percent of positions having a cost coming from a change in the initial cost given by S-UNIWARD.

Note that updating a cost causes a cost asymmetry since the cost of a $+1$ change is no longer equal to the cost of a -1 change, as in ASO. Besides, the update of the two costs

[14] An adversarial attack does not necessarily require us to use a deep learning classifier.

[15] In STC, before coding the message, the pixel positions of the image are shuffled thanks to the use of a pseudorandom shuffler, seeded by the secret stego-key. Note that this stego-key is shared between Alice and Bob. After the shuffling step, ADV-EMB selects the last β percent pixels of the *shuffled* image and modifies their associated cost and only those ones.

of a pixel is rather rough since it is a simple division by 2 for a direction ($+1$ or -1) and multiplication by 2 for the other direction. The sign of the gradient of loss, calculated by choosing the cover label, for a given pixel, makes it possible to determine for each of the two directions ($+1/-1$) if we reduce or increase the cost. The idea is as in ASO, to deceive the discriminant, since when we decide to reduce the value of a cost, it is to favor the direction of modification associated with this cost, and thus we promote getting closer to the cover class.

With such a scheme, security is improved. The fact that it is preferable to have a small number of modifications to the initial cost map probably makes it possible to preserve the initial embedding approach, and therefore not to introduce too many traces that could be detected by another steganalyzer [90]. That said, the update to the costs should probably be refined to better take into account the value of the gradient. For the moment, the selection of the β percent of pixels that will be modified is suboptimal, and this selection should eventually be done by looking at the initial cost of the whole pixel. Finally, as it is the case for ASO, if the discriminant is not powerful enough to carry out a steganalysis, then it can be totally counterproductive for Agent-Alice. Therefore there are many open questions regarding the convergence criterion, the stopping criterion, the number of iterations in the alternation between Agent-Alice and Agent-Eve, the definition of a metric for measuring the relevance of Agent-Eve, and so on. Note that an adversarial embedding *iterated* with Agent-Alice countering multiple versions of Agent-Eve has been proposed in IH&MMSec'2019 [94].

14.4.4 The family by 3-player game

The 3-player game concept is an extension of the previous family (see the family "adversarial-embedding *iterated*"), but this time with three agents and with all neural networks. Here the three agents Agent-Alice, Agent-Bob, and Agent-Eve are present. Note that Agent-Alice and Agent-Bob are "linked" since Agent-Bob is there only to add a constraint on the solution obtained by Agent-Alice. Thus the primary "game" is an antagonistic (or adversarial) game between Agent-Alice and Agent-Eve, whereas the "game" between Agent-Alice and Agent-Bob is rather cooperative, since these two agents share the common purpose of communicating (Agent-Alice and Agent-Bob both want Agent-Bob to be able to extract the message without errors). Fig. 14.14 from [95] summarizes the principle of the 3-player game. Agent-Alice takes a cover image, a message, and a stego-key, and after a discretization step generates a stego image. This stego image is used by Agent-Bob to retrieve a message. On the other side, Agent-Eve has to decide whether an image is cover or stego; this agent outputs a score.

Historically, after MOD and ASO, which only included two players, we can see the premise of the idea of three players appear in 2016 with the paper of Abadi and Andersen [96]. In this paper, Abadi and Andersen, from Google Brain, proposed a cryptographic toy-example for an encryption based on the use of three neural networks. The use of neural networks makes it easy to obtain a *strategic equilibrium* since the problem is expressed as a min–max problem, and its optimization can be carried out by the back-propagation

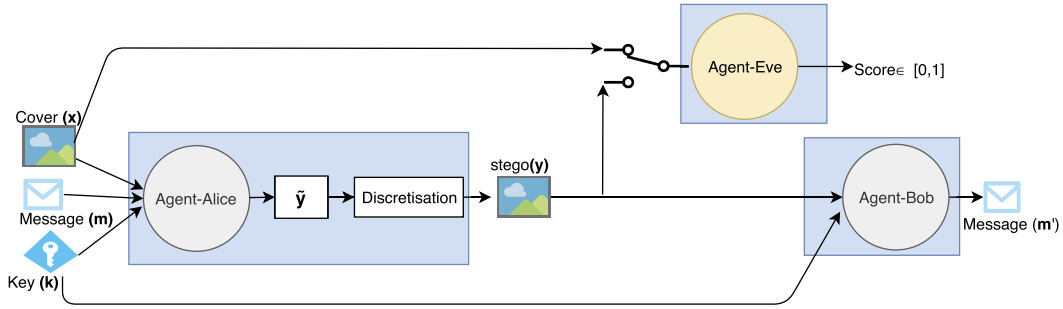

FIGURE 14.14 The overall architecture of the *3-player game*.

process. Naturally, this 3-player game concept can be transposed to steganography with the use of deep learning.

In December 2017 (GSIVAT; [97]) and in September 2018 (HiDDeN; [98]), two different teams from the machine learning community proposed, in NIPS'2017 and then in ECCV'2018, to achieve *strategic embedding* thanks to three CNNs, iteratively updated, playing the roles of Agent-Alice, Agent-Bob, and Agent-Eve. These two papers do not rigorously define the concept of the 3-player game, and there are erroneous assertions, mainly because the security and its evaluation are not correctly handled. If we place ourself in the standard framework to evaluate the empirical security of an embedding algorithm, that is, with a clairvoyant Eve, then the two approaches are very detectable. The most significant issues with these two papers are: first, neither of the two approaches uses a stego-key, which is the equivalent to always using the same key, and it leads to very detectable schemes [21]; second, there is no discretization of pixel values issued from Agent-Alice; third, the computational complexity, due to the use of fully connected blocks, leads to unpractical approaches; and fourth, the security evaluation is not carried out with a state-of-the-art steganalyzer.

At the beginning of 2019, Yedroudj et al. [95] redefined the 3-player concept by integrating the possibility of using a stego-key, treating the problem of discretization, going through convolution modules to have a scalable solution and using a suitable steganalyzer. The proposition is not comparable to classical adaptive embedding approaches, but there is a real potential to such an approach. The bit error rate is sufficiently small to be nullified, the embedding is done in the texture parts, and security could be improved in the future. As an example, the probability of error with a steganalysis by Yedroudj-Net [8] under equal errors prior, for a real payload size 0,3 bpp,[16] for images from BOWS2 database is 10.8%. This can, for example, be compared to the steganalysis of WOW [70] using the same conditions, which give a probability error of 22.4%. There is still a security gap, but this approach paves the way to much research. There are still open questions on the link

[16] A Hamming error correcting code ensures a null BER theoretically for most of the images, and thus a rate of 0.3 bpp for these images.

between Agent-Alice and Agent-Bob, on the use of GANs, the definition of losses, and the tuning of the compromises between the different constraints.

14.5 Conclusion

In this chapter, we practically completed a full presentation of the subject on deep learning in steganography and steganalysis since its appearance in 2015. We recalled the main elements of a CNN, and discussed the memory, time complexity, and practical problems for efficiency. Then, we explored the link between some past approaches sharing similarities with what is currently carried out in a CNN. Various networks until the beginning of 2019 with multiple scenarios are presented. Also, we touched on the recent approaches for steganography with deep learning. As mentioned in this chapter, many things have not been solved yet, and the major issue is to be able to experiment with more realistic hypotheses to be more "into the wild". The "holy grail" is cover-source mismatch and stego-mismatch, but in a way, the mismatch is a problem shared by the whole machine learning community. CNNs are now very present in the steganalysis community, and probably the next question is how to go a step further and produce clever networks? Finally, we think and hope that this chapter will help the community to understand what has been done and what are the next directions to explore.

References

[1] G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (Jul. 2006) 504–507.
[2] Y. Bengio, A.C. Courville, P. Vincent, Representation learning: a review and new perspectives, IEEE Transaction on Pattern Analysis and Machine Intelligence, PAMI 35 (8) (2013) 1798–1828.
[3] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (May 2015) 436–444.
[4] J. Fridrich, J. Kodovsky, Rich models for steganalysis of digital images, IEEE Transactions on Information Forensics and Security, TIFS 7 (3) (June 2012) 868–882.
[5] J. Kodovský, J. Fridrich, V. Holub, Ensemble classifiers for steganalysis of digital media, IEEE Transactions on Information Forensics and Security 7 (2) (2012) 432–444.
[6] Y. Qian, J. Dong, W. Wang, T. Tan, Deep learning for steganalysis via convolutional neural networks, in: Proceedings of Media Watermarking, Security, and Forensics 2015, MWSF'2015, Part of IS&T/SPIE Annual Symposium on Electronic Imaging, SPIE'2015, San Francisco, California, USA, vol. 9409, Feb. 2015, 94090J.
[7] S. Kouider, M. Chaumont, W. Puech, Adaptive steganography by oracle (ASO), in: Proceedings of the IEEE International Conference on Multimedia and Expo, ICME'2013, San Jose, California, USA, Jul. 2013, pp. 1–6.
[8] M. Yedroudj, F. Comby, M. Chaumont, Yedrouj-Net: an efficient CNN for spatial steganalysis, in: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'2018, Calgary, Alberta, Canada, Apr. 2018, pp. 2092–2096.
[9] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: Proceeding of Advances in Neural Information Processing Systems 25, NIPS'2012, Lake Tahoe, Nevada, USA, Curran Associates, Inc., Dec. 2012, pp. 1097–1105.
[10] G. Xu, H.Z. Wu, Y.Q. Shi, Structural design of convolutional neural networks for steganalysis, IEEE Signal Processing Letters 23 (5) (May 2016) 708–712.
[11] J. Ye, J. Ni, Y. Yi, Deep learning hierarchical representations for image steganalysis, IEEE Transactions on Information Forensics and Security, TIFS 12 (11) (Nov. 2017) 2545–2557.

[12] M. Yedroudj, M. Chaumont, F. Comby, How to augment a small learning set for improving the performances of a CNN-based steganalyzer?, in: Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2018, Burlingame, California, USA, 28 January – 2 February 2018, p. 7.
[13] M. Boroumand, M. Chen, J. Fridrich, Deep residual network for steganalysis of digital images, IEEE Transactions on Information Forensics and Security 14 (5) (May 2019) 1181–1193.
[14] A.D. Ker, P. Bas, R. Böhme, R. Cogranne, S. Craver, T. Filler, J. Fridrich, T. Pevný, Moving steganography and steganalysis from the laboratory into the real world, in: Proceedings of the 1st ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2013, Montpellier, France, Jun. 2013, pp. 45–58.
[15] P. Bas, T. Filler, T. Pevný, 'Break our steganographic system': the ins and outs of organizing BOSS, in: Proceedings of 13th International Conference on Information Hiding, IH'2011, in: Lecture Notes in Computer Science, vol. 6958, Springer, Prague, Czech Republic, May 2011, pp. 59–70.
[16] P. Bas, T. Furon, BOWS-2 contest (break our watermarking system), organised within the activity of the Watermarking Virtual Laboratory (Wavila) of the European Network of Excellence ECRYPT, 2008, organized between the 17th of July 2007 and the 17th of April 2008, http://bows2.ec-lille.fr/.
[17] K. He, X. Zhang, S. Ren, J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition, in: Proceedings of the European Conference on Computer Vision, ECCV'2014, Zurich, Switzerland, Sep. 2014, pp. 346–361.
[18] C.F. Tsang, J.J. Fridrich, Steganalyzing images of arbitrary size with CNNs, in: Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2018, Burlingame, California, USA, 28 January–2 February, 2018, 121.
[19] R. Zhang, F. Zhu, J. Liu, G. Liu, Depth-wise separable convolutions and multi-level pooling for an efficient spatial CNN-based steganalysis (previously named "efficient feature learning and multi-size image steganalysis based on CNN" on ArXiv), IEEE Transactions on Information Forensics and Security, TIFS (2020).
[20] S. Tan, B. Li, Stacked convolutional auto-encoders for steganalysis of digital images, in: Proceedings of Signal and Information Processing Association Annual Summit and Conference, APSIPA'2014, Chiang Mai, Thailand, Dec. 2014, pp. 1–4.
[21] L. Pibre, J. Pasquet, D. Ienco, M. Chaumont, Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover source-mismatch, in: Proceedings of Media Watermarking, Security, and Forensics, MWSF'2016, Part of I&ST International Symposium on Electronic Imaging, EI'2016, San Francisco, California, USA, Feb. 2016, pp. 1–11.
[22] G. Xu, H.-Z. Wu, Y.Q. Shi, Ensemble of CNNs for steganalysis: an empirical study, in: Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2016, Vigo, Galicia, Spain, Jun. 2016, pp. 103–107.
[23] J. Zeng, S. Tan, B. Li, J. Huang, Pre-training via fitting deep neural network to rich-model features extraction procedure and its effect on deep learning for steganalysis, in: Proceedings of Media Watermarking, Security, and Forensics 2017, MWSF'2017, Part of IS&T Symposium on Electronic Imaging, EI'2017, Burlingame, California, USA, Jan. 2017, p. 6.
[24] J. Zeng, S. Tan, B. Li, J. Huang, Large-scale JPEG image steganalysis using hybrid deep-learning framework, IEEE Transactions on Information Forensics and Security 13 (5) (May 2018) 1200–1214.
[25] M. Chen, V. Sedighi, M. Boroumand, J. Fridrich, JPEG-phase-aware convolutional neural network for steganalysis of JPEG images, in: Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2017, Drexel University, Philadelphia, PA, Jun. 2017, pp. 75–84.
[26] G. Xu, Deep convolutional neural network to detect J-UNIWARD, in: Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2017, Drexel University, Philadelphia, PA, Jun. 2017, pp. 67–73.
[27] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR'2016, Las Vegas, Nevada, Jun. 2016, pp. 770–778.
[28] X. Huang, S. Wang, T. Sun, G. Liu, X. Lin, Steganalysis of adaptive JPEG steganography based on ResDet, in: Proceedings of the Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA'2018, Hononulu, Hawaii, vol. 2018, Nov. 2018, pp. 12–15.

[29] T. Denemark, V. Sedighi, V. Holub, R. Cogranne, J. Fridrich, Selection-channel-aware rich model for steganalysis of digital images, in: Proceedings of IEEE International Workshop on Information Forensics and Security, WIFS'2014, Atlanta, Georgia, USA, Dec. 2014, pp. 48–53.
[30] T. Denemark, J.J. Fridrich, P.C. Alfaro, Improving selection-channel-aware steganalysis features, in: Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2016, San Francisco, California, USA, Feb. 2016, pp. 1–8.
[31] T. Denemark, M. Boroumand, J. Fridrich, Steganalysis features for content-adaptive JPEG steganography, IEEE Transactions on Information Forensics and Security 11 (8) (Aug. 2016) 1736–1746.
[32] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: Proceeding of International Conference on Learning Representations, ICLR'2015, San Diego, CA, May 2015, p. 12.
[33] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR'2015, Boston, MA, USA, Jun. 2015, pp. 1–9.
[34] B. Li, W. Wei, A. Ferreira, S. Tan, ReST-net: diverse activation modules and parallel subnets-based CNN for spatial image steganalysis, IEEE Signal Processing Letters 25 (5) (May 2018) 650–654.
[35] Y. Qian, J. Dong, W. Wang, T. Tan, Learning and transferring representations for image steganalysis using convolutional neural network, in: Proceedings of IEEE International Conference on Image Processing, ICIP'2016, Phoenix, Arizona, Sep. 2016, pp. 2752–2756.
[36] M. Chen, M. Boroumand, J.J. Fridrich, Deep learning regressors for quantitative steganalysis, in: Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2018, Burlingame, California, USA, 28 January–2 February, 2018, 160.
[37] J. Kodovský, J.J. Fridrich, Quantitative steganalysis using rich models, in: Proceeding of SPIE Media Watermarking, Security, and Forensics, Part of IS&T/SPIE 23th Annual Symposium on Electronic Imaging, SPIE Proceedings, SPIE'2013, San Francisco, California, USA, vol. 8665, Feb. 2013, pp. 1–11.
[38] A. Zakaria, M. Chaumont, G. Subsol, Quantitative and binary steganalysis in JPEG: a comparative study, in: Proceedings of the European Signal Processing Conference, EUSIPCO'2018, Roma, Italy, Sep. 2018, pp. 1422–1426.
[39] A. Zakaria, M. Chaumont, G. Subsol, Pooled steganalysis in JPEG: how to deal with the spreading strategy?, in: Proceedings of the IEEE International Workshop on Information Forensics and Security, WIFS'2019, Delft, the Netherlands, Dec. 2019, p. 6.
[40] B. Bayar, M.C. Stamm, A deep learning approach to universal image manipulation detection using a new convolutional layer, in: Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2016, Vigo, Galicia, Spain, Jun. 2016, pp. 5–10.
[41] F. Chollet, Xception: deep learning with depthwise separable convolutions, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, CVPR'2017, Honolulu, HI, USA, Jul. 2017, pp. 1800–1807.
[42] V. Holub, J. Fridrich, T. Denemark, Universal distortion function for steganography in an arbitrary domain, EURASIP Journal on Information Security, JIS 2014 (1) (2014).
[43] B. Li, M. Wang, J. Huang, X. Li, A new cost function for spatial image steganography, in: Proceedings of IEEE International Conference on Image Processing, ICIP'2014, Paris, France, Oct. 2014, pp. 4206–4210.
[44] B. Li, M. Wang, X. Li, S. Tan, J. Huang, A strategy of clustering modification directions in spatial image steganography, IEEE Transactions on Information Forensics and Security 10 (9) (2015) 1905–1917.
[45] D.P. Kingma, J.L. Ba, Adam: a method for stochastic optimization, in: Proceedings of Conference on Learning Representations, ICLR'2015, San Diego, CA, May 2015, p. 13.
[46] X. Deng, B. Chen, W. Luo, D. Luo, Fast and effective global covariance pooling network for image steganalysis, in: Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2019, Paris, France, Jul. 2019, pp. 230–234.
[47] V. Sedighi, J. Fridrich, Effect of imprecise knowledge of the selection channel on steganalysis, in: Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2015, Portland, Oregon, USA, 2015, pp. 33–42.
[48] V. Holub, J. Fridrich, Low-complexity features for JPEG steganalysis using undecimated DCT, IEEE Transactions on Information Forensics and Security, TIFS 10 (2) (Feb. 2015) 219–228.

[49] C. Xia, Q. Guan, X. Zhao, Z. Xu, Y. Ma, Improving GFR steganalysis features by using Gabor symmetry and weighted histograms, in: Proceedings of the 5th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2017, Philadelphia, Pennsylvania, USA, 2017, pp. 55–66.
[50] V. Holub, J. Fridrich, Phase-aware projection model for steganalysis of JPEG images, in: Proceedings of SPIE Media Watermarking, Security, and Forensics 2015, Part of IS&T/SPIE Annual Symposium on Electronic Imaging, SPIE'2015, San Francisco, California, USA, vol. 9409, Feb. 2015, p. 11.
[51] J. Huang, J. Ni, L. Wan, J. Yan, A customized convolutional neural network with low model complexity for JPEG steganalysis, in: Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2019, Paris, France, Jul. 2019, pp. 198–203.
[52] Q. Gibouloto, R. Cogranne, P. Bas, Steganalysis into the wild: How to define a source?, in: Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2018, Burlingame, California, USA, 28 January–2 February, 2018.
[53] D. Borghys, P. Bas, H. Bruyninckx, Facing the cover-source mismatch on JPHide using training-set design, in: Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2018, Innsbruck, Austria, Jun. 2018, pp. 17–22.
[54] R. Cogranne, Q. Giboulot, P. Bas, The ALASKA steganalysis challenge: a first step towards steganalysis, in: Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2019, Paris, France, Jul. 2019, pp. 125–137.
[55] G. Cancelli, G.J. Doërr, M. Barni, I.J. Cox, A comparative study of $+/-1$ steganalyzers, in: Proceedings of Workshop Multimedia Signal Processing, MMSP'2008, Cairns, Queensland, Australia, Oct. 2008, pp. 791–796.
[56] I. Lubenko, A.D. Ker, Steganalysis with mismatched covers: Do simple classifiers help?, in: Proceedings of the 14th ACM Multimedia and Security Workshop, MM&Sec'2008, MM&Sec'2012, Coventry, United Kingdom, Sep. 2012, pp. 11–18.
[57] I. Lubenko, A.D. Ker, Going from small to large data in steganalysis, in: Proceedings of Media Watermarking, Security, and Forensics III, Part of IS&T/SPIE 22th Annual Symposium on Electronic Imaging, SPIE'2012, San Francisco, California, USA, vol. 8303, Feb. 2012.
[58] J. Pasquet, S. Bringay, M. Chaumont, Steganalysis with cover-source mismatch and a small learning database, in: Proceedings of the 22nd European Signal Processing Conference, EUSIPCO'2014, Lisbon, Portugal, Sep. 2014, pp. 2425–2429.
[59] J. Butora, J.J. Fridrich, Detection of diversified stego sources with CNNs, in: Proceedings of Media Watermarking, Security, and Forensics, MWSF'2019, Part of IS&T International Symposium on Electronic Imaging, EI'2019, Burlingame, California, USA, Jan. 2019, 534.
[60] Y. Yousfi, J. Butora, J. Fridrich, Q. Giboulot, Breaking ALASKA: color separation for steganalysis in JPEG domain, in: Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2019, Paris, France, Jul. 2019, pp. 138–149.
[61] X. Li, X. Kong, B. Wang, Y. Guo, X. You, Generalized transfer component analysis for mismatched JPEG steganalysis, in: Proceedings of IEEE International Conference on Image Processing, ICIP'2013, Melbourne, Australia, Sep. 2013, pp. 4432–4436.
[62] X. Kong, C. Feng, M. Li, Y. Guo, Iterative multi-order feature alignment for JPEG mismatched steganalysis, Journal of Neurocomputing 214 (C) (Nov. 2016) 458–470.
[63] C. Feng, X.W. Kong, M. Li, Y. Yang, Y. Guo, Contribution-based feature transfer for JPEG mismatched steganalysis, in: Proceedings of IEEE International Conference on Image Processing, ICIP'2017, Beijing, China, Sep. 2017, pp. 500–504.
[64] D. Lerch-Hostalot, D. Megías, Unsupervised steganalysis based on artificial training sets, Engineering Applications of Artificial Intelligence 50 (C) (Apr. 2016) 45–59.
[65] M.A. Koçak, D. Ramirez, E. Erkip, D. Shasha, SafePredict: a meta-algorithm for machine learning that uses refusals to guarantee correctness, arXiv:1708.06425, 2017. [Online]. Available: http://arxiv.org/abs/1708.06425.
[66] G.J. Simmons, The subliminal channel and digital signatures, in: Proceeding of Crypto'83, Santa Barbara, CA, Plenum Press, New York, Aug. 1983, pp. 51–67.
[67] P. Schöttle, R. Böhme, A game-theoretic approach to content-adaptive steganography, in: Proceedings of the 14th International Conference on Information Hiding, IH'12, Berkeley, CA, USA, vol. 7692, 2012, pp. 125–141.

[68] P. Schöttle, R. Böhme, Game theory and adaptive steganography, IEEE Transactions on Information Forensics and Security 11 (4) (Apr. 2016) 760–773.
[69] T. Pevný, T. Filler, P. Bas, Using high-dimensional image models to perform highly undetectable steganography, in: Proceedings of the 12th International Conference on Information Hiding, IH'2010, Calgary, Alberta, Canada, in: Lecture Notes in Computer Science, vol. 6387, Springer, Jun. 2010, pp. 161–177.
[70] V. Holub, J. Fridrich, Designing steganographic distortion using directional filters, in: Proceedings of the IEEE International Workshop on Information Forensics and Security, WIFS'2012, Tenerife, Spain, Dec. 2012, pp. 234–239.
[71] V. Sedighi, R. Cogranne, J. Fridrich, Content-adaptive steganography by minimizing statistical detectability, IEEE Transactions on Information Forensics and Security, TIFS'2016 11 (2) (Feb. 2016) 221–234.
[72] T. Denemark, J. Fridrich, Improving steganographic security by synchronizing the selection channel, in: Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2015, Portland, Oregon, USA, 2015, pp. 5–14.
[73] L. Guo, J. Ni, Y.Q. Shi, An efficient JPEG steganographic scheme using uniform embedding, in: Proceedings of IEEE International Workshop on Information Forensics and Security, WIFS'2012, Costa Adeje, Tenerife, Spain, Dec. 2012, pp. 169–174.
[74] Y. Pan, J. Ni, W. Su, Improved uniform embedding for efficient JPEG steganography, in: Proceedings of the International Conference on Cloud Computing and Security, ICCCS 2016, Nanjing, China, in: Lecture Notes in Computer Science, vol. 10039, Springer, Jul. 2016, pp. 125–133.
[75] W. Li, W. Zhang, K. Chen, W. Zhou, N. Yu, Defining joint distortion for JPEG steganography, in: Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2018, Innsbruck, Austria, 2018, pp. 5–16.
[76] T. Filler, J. Fridrich, Design of adaptive steganographic schemes for digital images, in: Proceedings of SPIE Media Watermarking, Security, and Forensics, Part of IS&T/SPIE 21th Annual Symposium on Electronic Imaging, SPIE'2011, San Francisco Airport, California, United States, vol. 7880, Feb. 2011, 78800F.
[77] S. Kouider, M. Chaumont, W. Puech, Technical points about adaptive steganography by oracle (ASO), in: Proceedings of Signal Processing Conference, EUSIPCO'2012, 2012 Proceedings of the 20th European, Bucharest, Romania, Aug. 2012, pp. 1703–1707.
[78] T. Filler, J. Judas, J. Fridrich, Minimizing additive distortion in steganography using syndrome-trellis codes, IEEE Transactions on Information Forensics and Security 6 (3) (Sep. 2011) 920–935.
[79] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Proceedings of Advances in Neural Information Processing Systems, NIPS'2014, Dec. 2014, pp. 2672–2680.
[80] D. Volkhonskiy, I. Nazarov, B. Borisenko, E. Burnaev, Steganographic generative adversarial networks, 2017, unpublished.
[81] H. Shi, J. Dong, W. Wang, Y. Qian, X. Zhang, SSGAN: secure steganography based on generative adversarial networks, in: Proceedings of the 18th Pacific-Rim Conference on Multimedia, PCM'2017, Harbin, China, in: Lecture Notes in Computer Science, vol. 10735, Springer, Sep. 2017, pp. 534–544.
[82] V. Sedighi, J.J. Fridrich, R. Cogranne, Toss that BOSSbase, Alice!, in: Proceedings of Media Watermarking, Security, and Forensics, MWSF'2018, Part of IS&T International Symposium on Electronic Imaging, EI'2016, San Francisco, California, USA, Feb. 2016, pp. 1–9.
[83] J. Fridrich, Steganography in Digital Media, Cambridge University Press, 2009, Cambridge books online.
[84] D. Hu, L. Wang, W. Jiang, S. Zheng, B. Li, A novel image steganography method via deep convolutional generative adversarial networks, IEEE Access 6 (Jul. 2018) 38303–38314.
[85] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, in: Proceedings of the International Conference on Learning Representations, ICLR'2016, Caribe Hilton, San Juan, Puerto Rico, May 2016, p. 16.
[86] W. Quan, K. Wang, D.-M. Yan, X. Zhang, Distinguishing between natural and computer-generated images using convolutional neural networks, IEEE Transactions on Information Forensics and Security 13 (11) (Nov. 2018) 2772–2787.

[87] W. Tang, S. Tan, B. Li, J. Huang, Automatic steganographic distortion learning using a generative adversarial network, IEEE Signal Processing Letters 24 (10) (Oct. 2017) 1547–1551.
[88] J. Yang, D. Ruan, J. Huang, X. Kang, Y.-Q. Shi, An embedding cost learning framework using GAN (previously named "spatial image steganography based on generative adversarial network" on arXiv), IEEE Transactions on Information Forensics and Security, TIFS 15 (Jun. 2019) 839–851.
[89] J. Kodovský, J. Fridrich, On completeness of feature spaces in blind steganalysis, in: Proceedings of the 10th ACM Workshop on Multimedia and Security, MM&Sec'2008, Oxford, United Kingdom, 2008, pp. 123–132.
[90] J. Kodovsky, J. Fridrich, V. Holub, On dangers of overtraining steganography to incomplete cover model, in: Proceedings of the Thirteenth ACM Multimedia Workshop on Multimedia and Security, MM&Sec'2011, Buffalo, New York, USA, Sep. 2011, pp. 69–76.
[91] J. Yang, D. Ruan, X. Kang, Y.-Q. Shi, Towards automatic embedding cost learning for JPEG steganography, in: Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2019, Paris, France, Jul. 2019, pp. 37–46.
[92] Y. Zhang, W. Zhang, K. Chen, J. Liu, Y. Liu, N. Yu, Adversarial examples against deep neural network based steganalysis, in: Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2018, Innsbruck, Austria, Jun. 2018, pp. 67–72.
[93] W. Tang, B. Li, S. Tan, M. Barni, J. Huang, CNN-based adversarial embedding for image steganography, IEEE Transactions on Information Forensics and Security 14 (8) (Aug. 2019).
[94] S. Bernard, T. Pevný, P. Bas, J. Klein, Exploiting adversarial embeddings for better steganography, in: Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec'2019, Paris, France, Jul. 2019, pp. 216–221.
[95] M. Yedroudj, F. Comby, M. Chaumont, Steganography using a 3 player game, under submission, arXiv: 1907.06956, 2019. [Online]. Available: http://arxiv.org/abs/1907.06956.
[96] M. Abadi, D.G. Andersen, Learning to protect communications with adversarial neural cryptography, unpublished, arXiv:1610.06918, 2016. [Online]. Available: http://arxiv.org/abs/1610.06918.
[97] J. Hayes, G. Danezis, Generating steganographic images via adversarial training, in: Proceedings of Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems, NIPS'2017, Long Beach, CA, USA, Dec. 2017, pp. 1951–1960.
[98] J. Zhu, R. Kaplan, J. Johnson, L. Fei-Fei, HiDDeN: hiding data with deep networks, in: Proceedings of the 15th European Conference on Computer Vision, ECCV'2018, Munich, Germany, in: Lecture Notes in Computer Science, vol. 11219, Springer, Sep. 2018, pp. 682–697.